Interfejsy oszczędzania energii w jądrze Linuksa

Rafał J. Wysocki

Wydział Fizyki UW / Renesas Electronics / SUSE Labs

21 kwietnia 2012

Plan prezentacji

Wydajność dostosowana do potrzeb

Obecnie dąży się do tego, by systemy komputerowe zużywały jak najmniej energii – przy zachowaniu wydajności odpowiadającej potrzebom.

Wydajność dostosowana do potrzeb

Obecnie dąży się do tego, by systemy komputerowe zużywały jak najmniej energii – przy zachowaniu wydajności odpowiadającej potrzebom.

Wymagania tego rodzaju są brane pod uwagę przy projektowaniu sprzętu.

Wydajność dostosowana do potrzeb

Obecnie dąży się do tego, by systemy komputerowe zużywały jak najmniej energii – przy zachowaniu wydajności odpowiadającej potrzebom.

Wymagania tego rodzaju są brane pod uwagę przy projektowaniu sprzętu.

Niektóre rozwiązania mają charakter czysto sprzętowy, ale większość musi być uwzględniona w oprogramowaniu (aby były wykorzystywane).

Wydajność dostosowana do potrzeb

Obecnie dąży się do tego, by systemy komputerowe zużywały jak najmniej energii – przy zachowaniu wydajności odpowiadającej potrzebom.

Wymagania tego rodzaju są brane pod uwagę przy projektowaniu sprzętu.

Niektóre rozwiązania mają charakter czysto sprzętowy, ale większość musi być uwzględniona w oprogramowaniu (aby były wykorzystywane).

W części są to funkcje standardowe (ACPI, PCI, PCI Express, USB itp.), a w części są to pomysły poszczególnych producentów urządzeń.

Procesory

Stany P (*ang. P-states*)

Konfiguracje, w których częstotliwość taktowania procesora (rdzenia CPU) jest niższa od nominalnej (wymaga to dostosowania napięcia zasilającego).

Procesory

Stany P (*ang. P-states*)

Konfiguracje, w których częstotliwość taktowania procesora (rdzenia CPU) jest niższa od nominalnej (wymaga to dostosowania napięcia zasilającego).

Stany C (*ang. C-states*)

Stany, w których procesor (rdzeń CPU) nie wykonuje rozkazów i jest (zwykle częściowo) pozbawiony zasilania (i/lub referencyjnego sygnału zegarowego).

Procesory

Stany P (*ang. P-states*)

Konfiguracje, w których częstotliwość taktowania procesora (rdzenia CPU) jest niższa od nominalnej (wymaga to dostosowania napięcia zasilającego).

Stany C (*ang. C-states*)

Stany, w których procesor (rdzeń CPU) nie wykonuje rozkazów i jest (zwykle częściowo) pozbawiony zasilania (i/lub referencyjnego sygnału zegarowego).

Tryby „turbo” (*ang. turbo modes*)

Konfiguracje procesorów wielordzeniowych, w których częstotliwość taktowania części rdzeni jest zwiększana znacznie powyżej częstotliwości nominalnej kosztem pozostałych rdzeni (np. jeżeli są one w stanach C).

Urządzenia I/O

Stany o obniżonej częstotliwości taktowania (*ang. low-frequency*)

Analogiczne do stanów P dla procesorów.

Urządzenia I/O

Stany o obniżonej częstotliwości taktowania (*ang. low-frequency*)

Analogiczne do stanów P dla procesorów.

DVFS (*ang. Dynamic Voltage and Frequency Scaling*)

Sterowanie częstotliwością taktowania i napięciem zasilającym urządzeń.

Urządzenia I/O

Stany o obniżonej częstotliwości taktowania (*ang. low-frequency*)

Analogiczne do stanów P dla procesorów.

DVFS (*ang. Dynamic Voltage and Frequency Scaling*)

Sterowanie częstotliwością taktowania i napięciem zasilającym urządzeń.

OPP (*ang. Operation Preformance Points*)

Częstotliwości taktowania urządzeń I/O (i procesorów) w systemie mogą być skorelowane (tzn. w ogólności nie można ich obniżać dla każdego urządzenia z osobna). Wówczas dopuszczalne konfiguracje częstotliwości taktowania urządzeń w systemie są nazywane OPP.

Urządzenia I/O

Stany o obniżonej częstotliwości taktowania (*ang. low-frequency*)

Analogiczne do stanów P dla procesorów.

DVFS (*ang. Dynamic Voltage and Frequency Scaling*)

Sterowanie częstotliwością taktowania i napięciem zasilającym urządzeń.

OPP (*ang. Operation Preformance Points*)

Częstotliwości taktowania urządzeń I/O (i procesorów) w systemie mogą być skorelowane (tzn. w ogólności nie można ich obniżać dla każdego urządzenia z osobna). Wówczas dopuszczalne konfiguracje częstotliwości taktowania urządzeń w systemie są nazywane OPP.

Stany o obniżonym poborze mocy (*ang. low-power*)

Analogiczne do stanów C dla procesorów.

Domeny zegarowe i zasilania

Domena zegarowa (*ang. clock domain*)

Zbiór urzadzeń, dla których referencyjne sygnały zegarowe są ściśle skorelowane.

Domeny zegarowe i zasilania

Domena zegarowa (*ang. clock domain*)

Zbiór urzadzeń, dla których referencyjne sygnały zegarowe są ściśle skorelowane.

Wstrzymywanie zegara (*ang. clock gating*)

Technika polegająca na zmianie konfiguracji części synchronicznego układu scalonego w taki sposób, aby przerzutniki wchodzące w jej skład nie zmieniały swoich stanów.

Domeny zegarowe i zasilania

Domena zegarowa (*ang. clock domain*)

Zbiór urzadzeń, dla których referencyjne sygnały zegarowe są ściśle skorelowane.

Wstrzymywanie zegara (*ang. clock gating*)

Technika polegająca na zmianie konfiguracji części synchronicznego układu scalonego w taki sposób, aby przerzutniki wchodzące w jej skład nie zmieniały swoich stanów.

Domena zasilania (*ang. power domain*)

Zbiór urzadzeń o wspólnym zasilaniu. Odcięcie zasilania od domeny oznacza, że żadne z urządzeń w tej domenie nie będzie zasilane.

Platformy

Różne systemy mogą zawierać funkcjonalnie identyczne urządzenia w różnych konfiguracjach sprzętowych (np. w różnych zestawieniach, w różnych domenach zegarowych, w różnych domenach zasilania itp.).

Platformy

Różne systemy mogą zawierać funkcjonalnie identyczne urządzenia w różnych konfiguracjach sprzętowych (np. w różnych zestawieniach, w różnych domenach zegarowych, w różnych domenach zasilania itp.).

W jądrze Linuksa tego rodzaju zależności są reprezentowane przez platformy (*ang. platform*), np. OMAP 4 (TI), SH7372 (Renesas).

Platformy

Różne systemy mogą zawierać funkcjonalnie identyczne urządzenia w różnych konfiguracjach sprzętowych (np. w różnych zestawieniach, w różnych domenach zegarowych, w różnych domenach zasilania itp.).

W jądrze Linuksa tego rodzaju zależności są reprezentowane przez platformy (*ang. platform*), np. OMAP 4 (TI), SH7372 (Renesas).

Niektóre operacje muszą być przeprowadzane przez kod dostosowany do danej platformy (np. musi on „wiedzieć” do jakich domen zegarowych i zasilania należy urządzenie, co trzeba zrobić, żeby wstrzymać jego zegar, jaki jest zbiór OPP dostępnych dla systemu itp.).

Podsystemy

Podsystem (*ang. subsystem*)

Kategoria urządzeń związana z pewnym rodzajem standardowej obsługi (np. w zakresie konfiguracji).

Podsystemy

Podsystem (*ang. subsystem*)

Kategoria urządzeń związana z pewnym rodzajem standardowej obsługi (np. w zakresie konfiguracji).

Podsystemy w jądrze Linuksa

- Typy urządzeń (*ang. device types*).
- Klasy urządzeń (*ang. device classes*).
- Typy magistral (*ang. bus types*).

Podsystemy

Podsystem (*ang. subsystem*)

Kategoria urządzeń związana z pewnym rodzajem standardowej obsługi (np. w zakresie konfiguracji).

Podsystemy w jądrze Linuksa

- Typy urządzeń (*ang. device types*).
- Klasy urządzeń (*ang. device classes*).
- Typy magistral (*ang. bus types*).

Praktycznie nie ma potrzeby reprezentowania poszczególnych rodzajów podsystemów w różny sposób, także istnieje plan połączenia tych wszystkich reprezentacji w jedną ogólną.

Znaczenie podsystemów i platform

To, co można zrobić z urządzeniem, zależy od podsystemu, do którego ono należy oraz od platformy, na której działa jądro Linuksa.

Znaczenie podsystemów i platform

To, co można zrobić z urządzeniem, zależy od podsystemu, do którego ono należy oraz od platformy, na której działa jądro Linuksa.

Przeprowadzając operacje dotyczące urządzeń na ogół trzeba brać pod uwagę możliwości (*ang. capabilities*) charakterystyczne dla platformy i podsystemu.

Znaczenie podsystemów i platform

To, co można zrobić z urządzeniem, zależy od podsystemu, do którego ono należy oraz od platformy, na której działa jądro Linuksa.

Przeprowadzając operacje dotyczące urządzeń na ogół trzeba brać pod uwagę możliwości (*ang. capabilities*) charakterystyczne dla platformy i podsystemu.

Centralna część, czyli rdzeń (*ang. core*) jądra musi mieć możliwość inicjowania operacji związanych z oszczędzaniem energii w sposób niezależny od platformy i od tego do jakich podsystemów należą urządzenia.

Znaczenie podsystemów i platform

To, co można zrobić z urządzeniem, zależy od podsystemu, do którego ono należy oraz od platformy, na której działa jądro Linuksa.

Przeprowadzając operacje dotyczące urządzeń na ogół trzeba brać pod uwagę możliwości (*ang. capabilities*) charakterystyczne dla platformy i podsystemu.

Centralna część, czyli rdzeń (*ang. core*) jądra musi mieć możliwość inicjowania operacji związanych z oszczędzaniem energii w sposób niezależny od platformy i od tego do jakich podsystemów należą urządzenia.

Potrzebne są „standardowe” interfejsy (API) służące do tego celu.

CPU Power Management

CPUfreq (*ang. CPU frequency scaling*)

Automatyczne dostosowywanie częstotliwości taktownia i napięcia zasilającego do obciążenia procesora (o ile sprzęt ma takie możliwości). Wykorzystuje stany P (w ogólności DVFS).

CPU Power Management

CPUfreq (*ang. CPU frequency scaling*)

Automatyczne dostosowywanie częstotliwości taktownia i napięcia zasilającego do obciążenia procesora (o ile sprzęt ma takie możliwości). Wykorzystuje stany P (w ogólności DVFS).

CPUidle

Automatyczne wyłączanie procesorów (rdzeni CPU), które nie wykonują kodu należącego do jakiegokolwiek zadania. Jeśli nie są one wyłączane, to wykonują tzw. pętlę bezczynności (*ang. idle loop*).

CPU Power Management

CPUfreq (*ang. CPU frequency scaling*)

Automatyczne dostosowywanie częstotliwości taktownia i napięcia zasilającego do obciążenia procesora (o ile sprzęt ma takie możliwości). Wykorzystuje stany P (w ogólności DVFS).

CPUidle

Automatyczne wyłączanie procesorów (rdzeni CPU), które nie wykonują kodu należącego do jakiegokolwiek zadania. Jeśli nie są one wyłączane, to wykonują tzw. pętlę bezczynności (*ang. idle loop*).

CPUidle wykorzystuje stany C procesorów, o ile są one dostępne w sprzęcie. To, który stan C będzie wybrany dla bezczynnego rdzenia CPU (lub całego procesora), zależy od czasu, przez jaki będzie on bezczynny.

Oszczędzanie energii dla urządzeń I/O

Runtime PM API

Jednolity interfejs pozwalający na wykorzystywanie stanów o niskim poborze mocy z uwzględnieniem możliwości sterownika (*ang. driver*), podsystemu i platformy.

Oszczędzanie energii dla urządzeń I/O

Runtime PM API

Jednolity interfejs pozwalający na wykorzystywanie stanów o niskim poborze mocy z uwzględnieniem możliwości sterownika (*ang. driver*), podsystemu i platformy.

devfreq

Automatyczne dostosowywanie częstotliwości taktowania i napięcia zasilającego urządzeń I/O do obciążenia (odpowiednik CPUfreq). Wykorzystuje OPP.

Oszczędzanie energii dla urządzeń I/O

Runtime PM API

Jednolity interfejs pozwalający na wykorzystywanie stanów o niskim poborze mocy z uwzględnieniem możliwości sterownika (*ang. driver*), podsystemu i platformy.

devfreq

Automatyczne dostosowywanie częstotliwości taktowania i napięcia zasilającego urządzeń I/O do obciążenia (odpowiednik CPUfreq). Wykorzystuje OPP.

PM QoS (*ang. Power Management Quality of Service*)

Określanie ograniczeń dotyczących oszczędzania energii (jak bardzo agresywnie ma być oszczędzana energia).

Usypianie systemu

Usypianie systemu (*ang. system suspend*)

Przełączanie całego systemu do stanu o niskim poborze mocy, zwanego stanem uśpienia (*ang. sleep state*), w którym

- Procesory nie wykonują rozkazów (mogą być całkowicie lub częściowo wyłączone).
- Nie są przeprowadzane żadne operacje I/O z wyjątkiem tak zwanych zdarzeń budzących (*ang. wakeup events*).

Usypianie systemu

Usypianie systemu (*ang. system suspend*)

Przełączanie całego systemu do stanu o niskim poborze mocy, zwanego stanem uśpienia (*ang. sleep state*), w którym

- Procesory nie wykonują rozkazów (mogą być całkowicie lub częściowo wyłączone).
- Nie są przeprowadzane żadne operacje I/O z wyjątkiem tak zwanych zdarzeń budzących (*ang. wakeup events*).

Sygnał budzący (*ang. wakeup signal*)

Sygnał powodujący wyjście systemu ze stanu uśpienia. Po odebraniu go platforma powinna przełączyć co najmniej jeden procesor (rdzeń CPU) do stanu, w którym będzie on wykonywać rozkazy (począwszy od określonej lokacji w pamięci).

Hibernacja

Zawartość pamięci RAM jest kopiowana do urządzenia zachowującego zapis (*ang. non-volatile*), a następnie system jest wyłączany – z wyjątkiem urządzeń generujących sygnały budzące i części platformy pozwalającej na odbieranie tych sygnałów.

Hibernacja

Zawartość pamięci RAM jest kopiowana do urządzenia zachowującego zapis (*ang. non-volatile*), a następnie system jest wyłączany – z wyjątkiem urządzeń generujących sygnały budzące i części platformy pozwalającej na odbieranie tych sygnałów.

Różni się od usypiania systemu tym, że w stanie uśpienia zawartość pamięci RAM jest zachowywana, co wymaga zasilania.

Hibernacja

Zawartość pamięci RAM jest kopiowana do urządzenia zachowującego zapis (*ang. non-volatile*), a następnie system jest wyłączany – z wyjątkiem urządzeń generujących sygnały budzące i części platformy pozwalającej na odbieranie tych sygnałów.

Różni się od usypiania systemu tym, że w stanie uśpienia zawartość pamięci RAM jest zachowywana, co wymaga zasilania.

Zapis pamięci na „dysk” musi być przeprowadzany tak, aby po odczytaniu jej stan systemu był wewnętrznie spójny.

Hibernacja

Zawartość pamięci RAM jest kopiowana do urządzenia zachowującego zapis (*ang. non-volatile*), a następnie system jest wyłączany – z wyjątkiem urządzeń generujących sygnały budzące i części platformy pozwalającej na odbieranie tych sygnałów.

Różni się od usypiania systemu tym, że w stanie uśpienia zawartość pamięci RAM jest zachowywana, co wymaga zasilania.

Zapis pamięci na „dysk" musi być przeprowadzany tak, aby po odczytaniu jej stan systemu był wewnętrznie spójny.

Linux przeprowadza hibernację w dwóch etapach:

1. Tworzenie obrazu (*ang. image*).
2. Zapis obrazu.

CPUfreq

- Część centralna (*ang. core*) – sterowania działaniem, interfejs dla przestrzeni uzytkowników w *sysfs*.
- Moduły zarządzające (*ang. governors*) – podejmowanie decyzji dotyczących zmian konfiguracji na podstawie określonych kryteriów.
- Sterownik CPUfreq (*ang. CPUfreq driver*) – przeprowadzanie zmian konfiguracji na zlecenie modułów zarządzających.

CPUfreq

- Część centralna (*ang. core*) – sterowania działaniem, interfejs dla przestrzeni uzytkowników w *sysfs*.
- Moduły zarządzające (*ang. governors*) – podejmowanie decyzji dotyczących zmian konfiguracji na podstawie określonych kryteriów.
- Sterownik CPUfreq (*ang. CPUfreq driver*) – przeprowadzanie zmian konfiguracji na zlecenie modułów zarządzających.

- Lista modułów zarządzających zależy od konfiguracji jądra.

CPUfreq

- Część centralna (*ang. core*) – sterowania działaniem, interfejs dla przestrzeni uzytkowników w *sysfs*.
- Moduły zarządzające (*ang. governors*) – podejmowanie decyzji dotyczących zmian konfiguracji na podstawie określonych kryteriów.
- Sterownik CPUfreq (*ang. CPUfreq driver*) – przeprowadzanie zmian konfiguracji na zlecenie modułów zarządzających.

- Lista modułów zarządzających zależy od konfiguracji jądra.
- Moduł zarządzający jest wybierany poprzez *sysfs*.

CPUfreq

- Część centralna (*ang. core*) – sterowania działaniem, interfejs dla przestrzeni uzytkowników w *sysfs*.
- Moduły zarządzające (*ang. governors*) – podejmowanie decyzji dotyczących zmian konfiguracji na podstawie określonych kryteriów.
- Sterownik CPUfreq (*ang. CPUfreq driver*) – przeprowadzanie zmian konfiguracji na zlecenie modułów zarządzających.

- Lista modułów zarządzających zależy od konfiguracji jądra.
- Moduł zarządzający jest wybierany poprzez *sysfs*.
- Decyduje on o tym, co robić dalej. W przypadku podjęcia decyzji o zmianie częstotliwości wywołuje odpowiednią funkcję z części centralnej, która z kolei uruchamia odpowiednią metodę (*ang. callback*) ze sterownika.

Stany C i CPUidle

Własności stanów C procesorów (rdzeni CPU)

- Pobór mocy.
- Czas powrotu do pełnej funkcjonalności (*ang. wakeup latency*).
- Minimalny czas, na jaki jest sens przełączać procesor do tego stanu (*ang. target residency*).
- Zakres informacji, które trzeba odtworzyć podczas powrotu do pełnej funkcjonalności (np. zawartość schowków).

Stany C i CPUidle

Własności stanów C procesorów (rdzeni CPU)

- Pobór mocy.
- Czas powrotu do pełnej funkcjonalności (*ang. wakeup latency*).
- Minimalny czas, na jaki jest sens przełączać procesor do tego stanu (*ang. target residency*).
- Zakres informacji, które trzeba odtworzyć podczas powrotu do pełnej funkcjonalności (np. zawartość schowków).

Stany C są reprezentowane przez obiekty typu `struct cpuidle_state`, które tworzy sterownik CPUidle (*ang. CPUidle driver*).

Stany C i CPUidle

Własności stanów C procesorów (rdzeni CPU)

- Pobór mocy.
- Czas powrotu do pełnej funkcjonalności (*ang. wakeup latency*).
- Minimalny czas, na jaki jest sens przełączać procesor do tego stanu (*ang. target residency*).
- Zakres informacji, które trzeba odtworzyć podczas powrotu do pełnej funkcjonalności (np. zawartość schowków).

Stany C są reprezentowane przez obiekty typu `struct cpuidle_state`, które tworzy sterownik CPUidle (*ang. CPUidle driver*).

Każdy taki obiekt zawiera metodę (*ang. callback*) `.enter()` pozwalającą na wprowadzenie procesora do stanu C reprezentowanego przez ten obiekt.

Moduły zarządzające CPUidle

Dla każdego procesora (rdzenia CPU) rejestrowany jest obiekt typu `struct cpuidle_device` zawierający (między innymi) tablicę obiektów reprezentujących dostępne stany C.

Moduły zarządzające CPUidle

Dla każdego procesora (rdzenia CPU) rejestrowany jest obiekt typu `struct cpuidle_device` zawierający (między innymi) tablicę obiektów reprezentujących dostępne stany C.

Stan C, do którego należy przełączyć bezczynny procesor (rdzeń CPU) jest wybierany przez moduł zarządzający CPUidle (*ang. CPUidle governor*), udostępniający obiekt typu `struct cpuidle_governor`.

Moduły zarządzające CPUidle

Dla każdego procesora (rdzenia CPU) rejestrowany jest obiekt typu `struct cpuidle_device` zawierający (między innymi) tablicę obiektów reprezentujących dostępne stany C.

Stan C, do którego należy przełączyć bezczynny procesor (rdzeń CPU) jest wybierany przez moduł zarządzający CPUidle (*ang. CPUidle governor*), udostępniający obiekt typu `struct cpuidle_governor`.

Metoda `.select()` z tego obiektu jest wywoływana przez planistę CPU (*ang. CPU scheduler*) w przypadku stwierdzenia, że dany procesor (rdzeń CPU) jest bezczynny.

Moduły zarządzające CPUidle

Dla każdego procesora (rdzenia CPU) rejestrowany jest obiekt typu `struct cpuidle_device` zawierający (między innymi) tablicę obiektów reprezentujących dostępne stany C.

Stan C, do którego należy przełączyć bezczynny procesor (rdzeń CPU) jest wybierany przez moduł zarządzający CPUidle (*ang. CPUidle governor*), udostępniający obiekt typu `struct cpuidle_governor`.

Metoda `.select()` z tego obiektu jest wywoływana przez planistę CPU (*ang. CPU scheduler*) w przypadku stwierdzenia, że dany procesor (rdzeń CPU) jest bezczynny.

Może ona sprawdzić jak długo ten procesor (rdzeń CPU) ma być bezczynny i na tej podstawie wybiera docelowy stan C.

Role składników Runtime PM API

1. Podsystemy i sterowniki urządzeń udostępniają metody
   - `.runtime_suspend(dev)`
   - `.runtime_resume(dev)`
   - `.runtime_idle(dev)`

Role składników Runtime PM API

1. Podsystemy i sterowniki urządzeń udostępniają metody
   - `.runtime_suspend(dev)`
   - `.runtime_resume(dev)`
   - `.runtime_idle(dev)`
2. Podsystemy i sterowniki urządzeń obsługują sygnały budzące od urządzeń (*ang. remote wakeup*).

Role składników Runtime PM API

1. Podsystemy i sterowniki urządzeń udostępniają metody
   - `.runtime_suspend(dev)`
   - `.runtime_resume(dev)`
   - `.runtime_idle(dev)`
2. Podsystemy i sterowniki urządzeń obsługują sygnały budzące od urządzeń (*ang. remote wakeup*).
3. Część centralna (*ang. core*) zapewnia synchronizację (zapobieganie sytuacjom hazardowym itp.).

Role składników Runtime PM API

1. Podsystemy i sterowniki urządzeń udostępniają metody
   - `.runtime_suspend(dev)`
   - `.runtime_resume(dev)`
   - `.runtime_idle(dev)`
2. Podsystemy i sterowniki urządzeń obsługują sygnały budzące od urządzeń (*ang. remote wakeup*).
3. Część centralna (*ang. core*) zapewnia synchronizację (zapobieganie sytuacjom hazardowym itp.).
4. Bierze ona pod uwagę zależności między urządzeniami („rodzice” vs „dzieci”).

Role składników Runtime PM API

1. Podsystemy i sterowniki urządzeń udostępniają metody
   - `.runtime_suspend(dev)`
   - `.runtime_resume(dev)`
   - `.runtime_idle(dev)`
2. Podsystemy i sterowniki urządzeń obsługują sygnały budzące od urządzeń (*ang. remote wakeup*).
3. Część centralna (*ang. core*) zapewnia synchronizację (zapobieganie sytuacjom hazardowym itp.).
4. Bierze ona pod uwagę zależności między urządzeniami („rodzice" vs „dzieci").
5. Część centralna przeprowadza zliczanie odwołań (*ang. referece counting*) i udostępnia funkcje „pomocnicze" (*ang. helper functions*), poprzez które wywoływane są metody (gwarantują one zachowanie jednolitej konwencji wywoływania metod).

Konwencja wywoływania metod

Metody z podsystemów mogą być zastepowane przez metody z obiektów reprezentujących domeny zegarowe i/lub zasilania.

Konwencja wywoływania metod

Metody z podsystemów mogą być zastepowane przez metody z obiektów reprezentujących domeny zegarowe i/lub zasilania.

Konwencja wywoływania metod Runtime PM API (Linux v3.3 i nowsze)

1. Jeśli (dla danego urządzenia) istnieje obiekt reprezentujący domenę, wywoływana jest metoda z tego obiektu.
2. Jeśli istnieje obiekt reprezentujący typ urządzenia i udostępnia on metody Runtime PM, wywoływana jest metoda z tego obiektu.
3. Jeśli istnieje obiekt reprezentujący klasę urządzenia i udostępnia on metody Runtime PM, wywoływana jest metoda z tego obiektu.
4. Jeśli istnieje obiekt reprezentujący typ magistrali i udostępnia on metody Runtime PM, wywoływana jest metoda z tego obiektu.
5. W przeciwnym wypadku wywoływana jest metoda ze sterownika.

Etapy usypiania systemu (*ang. system suspend*)

Usypianie systemu jest operacją wieloetapową. Można w niej wyróżnić następujące kroki:

1. Wywoływanie funkcji powiadamiających (*ang. notifier*).
2. Zamrażanie procesów (*ang. freezig of tasks*).
3. Zawieszanie pracy urządzeń (*ang. device suspend*).
4. Wyłączanie procesorów (rdzeni CPU) – poza jednym.
5. Blokada obsługi przerwań i zawieszanie działania funkcji niskiego poziomu (*ang. system core suspend*).
6. Wyłączanie ostatniego aktywnego procesora (rdzenia CPU).

Etapy usypiania systemu (*ang. system suspend*)

Usypianie systemu jest operacją wieloetapową. Można w niej wyróżnić następujące kroki:

1. Wywoływanie funkcji powiadamiających (*ang. notifier*).
2. Zamrażanie procesów (*ang. freezig of tasks*).
3. Zawieszanie pracy urządzeń (*ang. device suspend*).
4. Wyłączanie procesorów (rdzeni CPU) – poza jednym.
5. Blokada obsługi przerwań i zawieszanie działania funkcji niskiego poziomu (*ang. system core suspend*).
6. Wyłączanie ostatniego aktywnego procesora (rdzenia CPU).

Podczas usypiania systemu działanie urządzeń I/O nie jest zawieszane za jednym zamachem. Czynność ta jest przeprowadzana w krokach zwanych „fazami” (*ang. phase*).

Zawieszanie pracy urządzeń podczas usypiania systemu

W jądrze Linuksa v3.3 (i wcześniej) zawieszanie pracy urządzeń jest przeprowadzane w 3 fazach: przygotowania (*ang. prepare*), wyłączania (*ang. suspend*) i wyłączania bez obsługi przerwań (*ang. suspend-no-interrupts*).

Zawieszanie pracy urządzeń podczas usypiania systemu

W jądrze Linuksa v3.3 (i wcześniej) zawieszanie pracy urządzeń jest przeprowadzane w 3 fazach: przygotowania (*ang. prepare*), wyłączania (*ang. suspend*) i wyłączania bez obsługi przerwań (*ang. suspend-no-interrupts*).

Do każdej z tych faz jest przypisana odpowiednia metoda w zestawie metod oszczędzania energii dla urządzeń (*ang. device PM callbacks*). Jest ona wywoływana dla wszystkich urządzeń w danej fazie.

Zawieszanie pracy urządzeń podczas usypiania systemu

W jądrze Linuksa v3.3 (i wcześniej) zawieszanie pracy urządzeń jest przeprowadzane w 3 fazach: przygotowania (*ang. prepare*), wyłączania (*ang. suspend*) i wyłączania bez obsługi przerwań (*ang. suspend-no-interrupts*).

Do każdej z tych faz jest przypisana odpowiednia metoda w zestawie metod oszczędzania energii dla urządzeń (*ang. device PM callbacks*). Jest ona wywoływana dla wszystkich urządzeń w danej fazie.

Jeśli metoda wywołana w danej fazie dla jednego z urządzeń zwróci błąd, to cała operacja jest przerywana i system wraca do stanu aktywności.

Zawieszanie pracy urządzeń podczas usypiania systemu

W jądrze Linuksa v3.3 (i wcześniej) zawieszanie pracy urządzeń jest przeprowadzane w 3 fazach: przygotowania (*ang. prepare*), wyłączania (*ang. suspend*) i wyłączania bez obsługi przerwań (*ang. suspend-no-interrupts*).

Do każdej z tych faz jest przypisana odpowiednia metoda w zestawie metod oszczędzania energii dla urządzeń (*ang. device PM callbacks*). Jest ona wywoływana dla wszystkich urządzeń w danej fazie.

Jeśli metoda wywołana w danej fazie dla jednego z urządzeń zwróci błąd, to cała operacja jest przerywana i system wraca do stanu aktywności.

W jądrze Linuksa v3.4-rc1 została dodana jeszcze jedna faza zawieszania pracy urządzeń – opóźnione wyłączanie (*ang. late suspend*).

Urządzenia budzące

W (systemowym) stanie uśpienia urządzenia I/O mogą być całkowicie pozbawione zasilania, poza urządzeniami skonfigurowanymi do generowania sygnałów budzących, zwanymi urządzeniami budzącymi (*ang. wakeup devices*).

Urządzenia budzące

W (systemowym) stanie uśpienia urządzenia I/O mogą być całkowicie pozbawione zasilania, poza urządzeniami skonfigurowanymi do generowania sygnałów budzących, zwanymi urządzeniami budzącymi (*ang. wakeup devices*).

Urządzenia budzące muszą być zasilane w stopniu wystarczającym do wygenerowania sygnału budzącego.

Urządzenia budzące

W (systemowym) stanie uśpienia urządzenia I/O mogą być całkowicie pozbawione zasilania, poza urządzeniami skonfigurowanymi do generowania sygnałów budzących, zwanymi urządzeniami budzącymi (*ang. wakeup devices*).

Urządzenia budzące muszą być zasilane w stopniu wystarczającym do wygenerowania sygnału budzącego.

Nie każde urządzenie I/O może być urządzeniem budzącym. O tym, które z urządzeń mających takie możliwości mają być skonfigurowane jako urządzenia budzące, decydują procesy z przestrzeni użytkowników.

Urządzenia budzące

W (systemowym) stanie uśpienia urządzenia I/O mogą być całkowicie pozbawione zasilania, poza urządzeniami skonfigurowanymi do generowania sygnałów budzących, zwanymi urządzeniami budzącymi (*ang. wakeup devices*).

Urządzenia budzące muszą być zasilane w stopniu wystarczającym do wygenerowania sygnału budzącego.

Nie każde urządzenie I/O może być urządzeniem budzącym. O tym, które z urządzeń mających takie możliwości mają być skonfigurowane jako urządzenia budzące, decydują procesy z przestrzeni użytkowników.

Istnieje problem obsługi sygnałów z urządzeń budzących podczas usypiania systemu.

Przywracanie aktywności systemu (*ang. system resume*)

Operacja ta polega na odwróceniu skutków czynności przeprowadzonych podczas usypiania systemu.

Przywracanie aktywności systemu (*ang. system resume*)

Operacja ta polega na odwróceniu skutków czynności przeprowadzonych podczas usypiania systemu.

W trakcie przywracania aktywności systemu praca urządzeń jest przywracana w fazach, w analogii do zawieszania ich pracy podczas usypiania systemu.

Przywracanie aktywności systemu (*ang. system resume*)

Operacja ta polega na odwróceniu skutków czynności przeprowadzonych podczas usypiania systemu.

W trakcie przywracania aktywności systemu praca urządzeń jest przywracana w fazach, w analogii do zawieszania ich pracy podczas usypiania systemu.

W jądrze Linuksa v3.3 (i wcześniej) są to 3 fazy: przywracanie pracy bez obsługi przerwań (*ang. resume-no-interrupts*), przywracanie pracy (*ang. resume*) i czynności końcowe (*ang. complete*).

Przywracanie aktywności systemu (*ang. system resume*)

Operacja ta polega na odwróceniu skutków czynności przeprowadzonych podczas usypiania systemu.

W trakcie przywracania aktywności systemu praca urządzeń jest przywracana w fazach, w analogii do zawieszania ich pracy podczas usypiania systemu.

W jądrze Linuksa v3.3 (i wcześniej) są to 3 fazy: przywracanie pracy bez obsługi przerwań (*ang. resume-no-interrupts*), przywracanie pracy (*ang. resume*) i czynności końcowe (*ang. complete*).

W jądrze Linuksa v3.4-rc1 została dodana jeszcze jedna faza przywracania pracy urządzeń – wczesne przywracanie pracy (*ang. early resume*).

Etapy hibernacji

1. Wywoływanie funkcji powiadamiających (*ang. notifier*).
2. Zamrażanie procesów (*ang. freezig of tasks*).
3. Zamrażanie urządzeń (*ang. device freeze*).
4. Wyłączanie procesorów (rdzeni CPU) – poza jednym.
5. Blokada obsługi przerwań i zawieszanie działania funkcji niskiego poziomu.
6. Tworzenie obrazu systemu do zachowania (*ang. hibernation image*).
7. Przywracanie działania funkcji niskiego poziomu (*ang. system core resume*) i odblokowanie obsługi przerwań.
8. Włączanie procesorów (rdzeni CPU).
9. Odmrażanie urządzeń (*ang. device thaw*).
10. Zapisywanie obrazu systemu.
11. Wyłączanie urządzeń (*ang. device power off*).
12. Wyłączanie procesorów (rdzeni CPU) – poza jednym.
13. Blokada obsługi przerwań i zawieszanie działania funkcji niskiego poziomu.
14. Wyłączanie systemu.

Obsługa urządzeń podczas hibernacji

Metody związane z hibernacją są wywoływane w fazach, podobnie jak dla usypiania systemu, ale podczas hibernacji ma miejsce więcej przejść systemu z jednego stanu do drugiego.

Obsługa urządzeń podczas hibernacji

Metody związane z hibernacją są wywoływane w fazach, podobnie jak dla usypiania systemu, ale podczas hibernacji ma miejsce więcej przejść systemu z jednego stanu do drugiego.

Analogiczna sytuacja ma miejsce podczas przywracania aktywności systemu po załadowaniu zawartości obrazu do pamięci.

Obsługa urządzeń podczas hibernacji

Metody związane z hibernacją są wywoływane w fazach, podobnie jak dla usypiania systemu, ale podczas hibernacji ma miejsce więcej przejść systemu z jednego stanu do drugiego.

Analogiczna sytuacja ma miejsce podczas przywracania aktywności systemu po załadowaniu zawartości obrazu do pamięci.

Zamrażanie : Przygotowanie do tworzenia (lub odczytywania) obrazu.

Odmrażanie : Przywracanie funkcjonalności po utworzeniu obrazu.

Wyłączanie : System przestaje pracować.

Odtwarzanie : Przywracanie pracy po odczytaniu obrazu.

Obsługa urządzeń podczas hibernacji – fazy

W jądrze Linuksa v3.3 (i wcześniej) każde z wymienionych przejść (dla urządzeń) ma fazę „normalną” i fazę „bez obsługi przerwań” (fazy przygotowania i czynności końcowych są wspólne).

Obsługa urządzeń podczas hibernacji – fazy

W jądrze Linuksa v3.3 (i wcześniej) każde z wymienionych przejść (dla urządzeń) ma fazę „normalną” i fazę „bez obsługi przerwań” (fazy przygotowania i czynności końcowych są wspólne).

W jądrze Linuksa v3.4-rc1 zostały dodane fazy:

- opóźnionego zamrażania (*ang. late freeze*)
- wczesnego odmrażania (*ang. early thaw*)
- opóźnionego wyłączania (*ang. late power off*)
- wczesnego odtwarzania (*ang. early restore*)

Obsługa urządzeń podczas hibernacji – fazy

W jądrze Linuksa v3.3 (i wcześniej) każde z wymienionych przejść (dla urządzeń) ma fazę „normalną” i fazę „bez obsługi przerwań” (fazy przygotowania i czynności końcowych są wspólne).

W jądrze Linuksa v3.4-rc1 zostały dodane fazy:

- opóźnionego zamrażania (*ang. late freeze*)
- wczesnego odmrażania (*ang. early thaw*)
- opóźnionego wyłączania (*ang. late power off*)
- wczesnego odtwarzania (*ang. early restore*)

Wprowadzenie nowych faz obsługi urządzeń podczas usypiania systemu i hibernacji (oraz przywracania jego aktywności) ma na celu umożliwienie wykorzystania tych samych metod (procedur) do obsługi Runtime PM API.

Przywracanie aktywności systemu po hibernacji

Bardziej skomplikowane od przywracania aktywności systemu po uśpieniu go, ale w zasadzie przebiega analogicznie.

Przywracanie aktywności systemu po hibernacji

Bardziej skomplikowane od przywracania aktywności systemu po uśpieniu go, ale w zasadzie przebiega analogicznie.

Podczas przywracania aktywności systemu po hibernacji mamy do czynienia z dwiema (*ang. różnymi*) instancjami jądra: instancją odczytującą obraz (*ang. boot kernel*) oraz instancją „zahibernowaną” (*ang. image kernel*).

Przywracanie aktywności systemu po hibernacji

Bardziej skomplikowane od przywracania aktywności systemu po uśpieniu go, ale w zasadzie przebiega analogicznie.

Podczas przywracania aktywności systemu po hibernacji mamy do czynienia z dwiema (*ang. różnymi*) instancjami jądra: instancją odczytującą obraz (*ang. boot kernel*) oraz instancją „zahibernowaną” (*ang. image kernel*).

Podczas tego procesu instancja odczytująca obraz jest zastępowana przez instancję „zahibernowaną” (po odczytaniu obrazu wykonywany jest skok w odpowiednie miejsce kodu „zahibernowanego” jądra).

Przywracanie aktywności systemu po hibernacji

Bardziej skomplikowane od przywracania aktywności systemu po uśpieniu go, ale w zasadzie przebiega analogicznie.

Podczas przywracania aktywności systemu po hibernacji mamy do czynienia z dwiema (*ang. różnymi*) instancjami jądra: instancją odczytującą obraz (*ang. boot kernel*) oraz instancją „zahibernowaną” (*ang. image kernel*).

Podczas tego procesu instancja odczytująca obraz jest zastępowana przez instancję „zahibernowaną” (po odczytaniu obrazu wykonywany jest skok w odpowiednie miejsce kodu „zahibernowanego” jądra).

Instancja odczytująca obraz może jednak w dowolny sposób skonfigurować urządzenia przed przekazaniem kontroli do instancji „zahibernowanej”.

Pytania?

Pytania?

Dziękuję za uwagę!

Bibliografia

J. Corbet, *The cpuidle subsystem* (http://lwn.net/Articles/384146/).

R. J. Wysocki, *Why We Need More Device Power Management Callbacks* (https://events.linuxfoundation.org/images/stories/pdf/lfcs2012_wysocki.pdf).

Dokumentacja i kod źródłowy

- Documentation/cpu-freq/*
- Documentation/cpuidle/*
- Documentation/power/devices.txt
- Documentation/power/pci.txt
- Documentation/power/runtime_pm.txt
- include/linux/cpufreq.h
- include/linux/cpuidle.h
- include/linux/device.h
- include/linux/pm.h
- include/linux/pm_runtime.h
- include/linux/suspend.h
- drivers/acpi/processor_idle.c
- drivers/base/power/*
- drivers/cpufreq/*
- drivers/cpuidle/*
- kernel/power/*